Name: *Student id:* *Section.* *Serial#:*

QUESTION #	1	2	3	4	TOTAL
MAX POINTS	20	20	20	24	
POINTS EARNED	19	19.5	16	23.5	78

UNIVERSITY OF BAHRAIN — COLLEGE OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE — TIME: 90 MINUTES
ITCS242: ASSEMBLY LANGUAGE PROGRAMMING — SECOND TEST — DATE: MAY 21, 2013

****

QUESTION ONE: **{20 pts}**

1) Write the needed instructions to store in AL and AH the counts of ONE and ZERO bits in a predefined memory location X of size double word. **Keep the value in X unchanged.**

10

```
        mov ax,0
        mov ecx,32
        CLC?
LF:     ROL X,1
        Jc L1          ; jump if carry =1
        inc AH
        jmp L2
L1:     inc AL
L2:     Loop LF
```


2) Given: `k dword ?` ; where k is an integer value between 4 and 16; Write the needed instructions to draw a triangle of letters consisting of K rows as shown below. For K=6 a triangle is:

9

```
        mov esi,1
        mov edi,0
        mov ecx,k
LF:     push ecx
        mov ecx, esi
LM:     Mov AL,'A'+edi
        CALL writeChar
        inc edi
        Loop LM
        pop ecx
        mov edi,0   ??
        CALL Crlf
        inc esi
        Loop LF
```

A
AB
ABC
ABCD
ABCDE
ABCDEF

Name: *Student id:* *Section:1* *Serial#:*

QUESTION TWO: Write a sequence of assembly instructions to perform each of the following tasks:

1) Give no more than 5 instructions to inverse even-numbered bits in **eflags** register. Keep other bits unchanged. **{5 pts}**

4.5

```
pushfd
pop  eax
xor  eax, 55555555h
push eax
popfd
```

3 2 1 0
0 1 0 1
555

2) Give no more than 3 instructions to inverse all bits in the last **dword** pushed onto the stack. **{3 pts}**

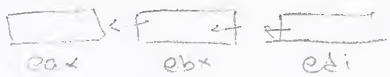
3

```
pop  eax
xor  eax,0FFFFFFFFh
push eax
```

3) Give no more than 5 instructions to shift the entire value in **EAX:EBX:EDI** 8 bits to the left. **{5 pts}**

5

```
mov  ecx, 8
LO:SHL  edi,1
RCL  ebx,1
RCL  eax,1
Loop LO
```



4) Give no more than 3 instructions to divide the signed predefined words values U9 / U8 **{3 pts}**

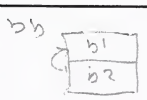
3

```
mov  ax, U9
CWD
Div  U8
```

5) Give no more than 4 instructions to store in bb the product of multiplying the 2 bytes of bb. **{4 pts}**

4

```
mov  AL, byte ptr bb
mul  byte ptr bb+1
mov  bb, ax
```



19.5

QUESTION THREE: What will be in the specified registers after executing the following code? **{ 20 pts}**

a)
```
MOV   AX, 6C50H
MOV   BX, 0EC4FH
IMUL  BH
```
AX = EE40 H

b)
```
MOV   AX, 2A49H
MOV   CX, 2090H
IDIV  CL
```
AX = 49A0 H

c)
```
MOV   AX, 0B9A5H
TEST  AX, 0EC79H
XOR   AX, 0AD7AH
```
AX = 140F H

d)
```
MOV   BX, 794CH
MOV   AX, 2F08H
SHRD  AX, BX, 4
```
AX = C2F0 H

e)
```
MOV   AX,3AFBH
MOV   BX,7A36H
AND   AX, BX
ROL   AX, 8
RCR   BX, 1
```
AX = 323A H

BX = 3D1B H

f)
```
MOV   AX,4E79H
MOV   BX,7A36H
MOV   CL, 2
SAR   AX, CL
NEG   BX
```
AX = 239E H

BX = 85CA H

g)
```
MOV   AX,3FFFH
MOV   BX,6940H
CMP   AL, AH
JL    L1
INC   BL
JMP   L2
L1:INC  BH
L2:
```
BX = 6940 H

h)
```
MOV   AX,3FFFH
MOV   BX,5A2DH
CMP   AL, AH
JB    L3
DEC   BL
JMP   L4
L3:DEC  BH
L4:
```
BX = 592D H

16

Name: *Student id:* *Section:1* *Serial#:*

QUESTION FOUR: Write an Assembly program that defines in the data segment three arrays `UU`, `GG`, and `RR` each consisting of 24 values. The types of `UU` and `GG` are signed bytes with values randomly generated in procedure **`tasks`**. The program consists of two procedures described as follows: **{24 pts}**

a) The first procedure named **FUN** accepts 3 parameters: ***x, y*** (variables of type signed byte) and ***f*** is a pointer to a variable. The purpose of the procedure **FUN** is to calculate the value of ***f*** as follows: ***f = 32[(x * y)/(x + y)].*** (Define a function **FUN** in a form that allows using ***invoke***).

b) The second procedure named **`tasks`** performs the following:
   1) Randomly generates 24 values `(-20 ... +20)` and store them in array `UU`.
   2) Randomly generates 24 values `(-30 ... +65)` and store them in array `GG`.
   3) Calls the procedure **FUN** to calculate the value of `RR[j]= f(UU[j], GG[j])` for the corresponding values of `UU[j]` and `GG[j]`.

```
Include Irvin32.inc
.data
  UU sbyte 24 dup(?)
  GG sbyte 24 dup(?)
  RR sdword 24 dup(?)

.code
FUN proc uses eax ebx ecx edx, x:sbyte, y:sbyte, f:ptr sdword
  mov AL, x
  imul y          ; Ax= x*y
  movsx bx, x
  movsx dx, y
  add bx, dx      ;bx= x+y
  CWD             ; Ax --> dx:Ax
  idiv bx         ;  ax= (x*y) / (x+y)
  movsx eax, ax
  SHL eax, 5      ; resul <- eax
  mov ebx, f
  mov [ebx], eax
  ret
FUN endp
tasks proc
  CALL Randomize
  mov esi, 0
  mov ecx, lengthof UU
L1: mov eax, 41
  CALL RandomRange
  sub eax, 20
  mov UU[esi], AL
  mov eax, 96
  CALL RandomRange
  sub eax, 30
  mov GG[esi], AL
  inc esi
  Loop L1
  mov edi, 0
  mov esi, 0
  mov ecx, lengthof RR
LN: invoke FUN, UU[esi], GG[esi],
                    addr RR[edi]
  inc esi
  add edi, 4
  Loop LN
  Exit
tasks endp
  end tasks.
```

-0.5 SAL??

23.5